

WHAT IS CLAIMED IS:

1. A message queue management system configured to manage a plurality of queues including a first message queue of a first data store type and a second message queue of a second data store type, wherein the first message queue and the second message queue exhibit consistent behavior across a plurality of computer mediums, the message queue management system comprising:

a journalling module configured to track and manage message states of messages stored in the first message queue and the second message queue including a journalled state whereby journalled messages are retained in the first message queue and the second message queue;

a once-only message processing module configured to track a plurality of message states to prevent re-processing of previously processed messages stored in the first message queue and the second message queue;

an interoperability module configured to coordinate interactions among the plurality of queues enabling a distributed workflow;

a recovery module configured to perform data recovery on the plurality of queues upon the event of a failure;

a central managing module configured to provide single point control of the plurality of queues, wherein the single point control includes controlling a plurality of publishers and a plurality of subscribers; and

a load balancing module configured to designate a subset of the plurality of subscribers to at least the first message queue in a subscriber pool such that messages in the first message queue are processed by a first available subscriber in the subscriber pool.

2. The message queue management system of Claim 1, wherein the load balancing module is further configured to enable the subscribers to subscribe to at least the first message queue in parallel with subscribers not in the subscriber pool.

3. A message queuing system for computer networks that provides reliable communication, ensures non-repetitive delivery, and is implemented in a medium-independent environment, the message queuing system comprising:

a message queue configured to run on one of a plurality of environments and to store messages received from message publishers and monitor retrieval by a plurality of message subscribers wherein a message is retained in the message queue after at least one of the plurality of message subscribers has accessed the message to permit additional access to the message;

a message state module configured to track message states of a message wherein message state information indicates whether the message has been made available to and retrieved by at least one of the plurality of the message subscribers;

a recording module configured to selectively retain the message state information related to corresponding messages at least until the message is removed from the message queue; and

an interface module configured to receive and send a plurality of types of data objects in messages and to communicate with a plurality of types of data messaging systems that act as message publishers or message subscribers.

4. The message queuing system of Claim 3, wherein at least one of the plurality of the data messaging systems acts as both a message publisher and a message subscriber.

5. The message queuing system of Claim 3, wherein the message queue is further configured to provide load balancing.

6. The message queuing system of Claim 5, wherein the load balancing is implemented using subscriber pooling.

7. The message queuing system of Claim 3, wherein the message state information is used to ensure that messages are acquired in the designated order by at least one of the plurality of message subscribers.

8. The message queuing system of Claim 3, wherein a message is intended for a single message subscriber.

9. The message queuing system of Claim 3, wherein a message is intended for a plurality of message subscribers.

10. The message queuing system of Claim 3, wherein the message is deleted from the message queue during a designated clean up process.

11. The message queuing system of Claim 3, wherein the message is retained in the message queue to assist in recovery of messages during a system failure.

12. The message queuing system of Claim 3, wherein the message is retained in the message queue to assist in at least one of archiving of the message, manual distribution of the message, editing of the message, or resending of the message.

13. The message queuing system of Claim 3, wherein a message is intended for a plurality of message subscribers.

14. The message queuing system of Claim 3, wherein the message state information includes message publisher state information and corresponding message subscriber state information.

15. The message queuing system of Claim 14, wherein the message state information is stored within the message.

16. The message queuing system of Claim 3, wherein the message states reflect the message publisher's status of the message in relation to the intended message subscribers and an individual message subscriber's status of the message in relation to the corresponding individual message subscriber.

17. The message queuing system of Claim 3, wherein the message state includes a sequence number, publisher state information, and subscriber state information.

18. The message queuing system of Claim 17, wherein the publisher state information includes information relating to an "unrevealed" state, information relating to a "revealed" state, information relating to an "unavailable" state, and information relating to a "journalled" state.

19. The message queuing system of Claim 17, wherein the subscriber state information includes information relating to a "get" state, information relating to a "fetched" state, information relating to an "unusable" state, information relating to an "expired" state, information relating to an "inprocess" state, information relating to a "done" state, and information relating to a "journalled" state.

20. The message queuing system of Claim 3, wherein the message queue is configured to interface with a plurality of different database protocols.

21. The message queuing system of Claim 3, wherein at least one of the types of a plurality of data messaging systems is a database.

22. The message queuing system of Claim 3, wherein at least one of the types of a plurality of data messaging systems is a flat file directory.

5 23. The message queuing system of Claim 3, wherein at least one of the types of a plurality of data messaging systems is a second message queuing system.

24. The message queuing system of Claim 3, wherein at least one of a plurality of data objects is a database record.

10 25. The message queuing system of Claim 3, wherein at least one of a plurality of data objects is an electronic mail message.

26. The message queuing system of Claim 3, wherein at least one of a plurality of data objects is a web page.

27. A method of message queuing among a plurality of message publishers and a plurality of message subscribers, the method comprising:

15 receiving a data object from a message publisher intended for at least one message subscriber;

processing the data object to create a corresponding message wherein the corresponding message includes the data object and message information;

storing the message in a message queue;

20 tracking the status of the message from the perspective of the message publisher;

tracking the status of the message from the perspective of the at least one message subscriber; and

25 selectively retaining the message to remain in the message queue after the message has been retrieved by the at least one message subscriber.

28. The method of Claim 27, wherein the message queue is configurable to run on a plurality of platforms.

29. The method of Claim 27, wherein the message queue is configurable to provide load balancing.

30 30. The method of Claim 27, wherein the message queue is configurable to provide subscriber pooling.

31. The method of Claim 27, wherein the message information includes a message identifier and a message header.

32. The method of Claim 28, wherein the message header includes message state information.

5 33. The method of Claim 28, wherein the message header includes message state information for the message publisher and the at least one message subscriber.

34. The method of Claim 33, wherein the message information for the message publisher includes information related to message publisher states wherein the message publisher states include a "put" state, an "unavailable" state, a "revealed" state, and a "journalled" state.

35. The method of Claim 33, wherein the message information for the at least one message subscriber includes information related to message subscriber states wherein the message subscriber states include a "get" state, a "fetched" state, an "unusable" state, an "expired" state, an "inprocess" state, a "done" state, and a "journalled" state.

36. The method of Claim 32, wherein the message state information is stored in a log that includes time stamps for the corresponding message states.

37. A method for journalling a plurality of messages in a message queue, wherein the plurality of messages include a plurality of corresponding message states, the method comprising:

receiving a message, wherein the message includes a publisher message state and a subscriber message state;
placing the message in a message queue;
changing the publisher message state of the message to a journalled state;
25 and
retaining the message in the message queue.

38. The method of Claim 37 further comprising changing the subscriber message state of the message to a terminal state to prevent re-processing of the message.

39. A method of subscriber pooling a plurality of messages in a message queue, wherein the plurality of message includes corresponding publisher states, the method comprising:

placing a message in a message queue;
informing a plurality of subscribers that the message is available; and
upon retrieval by one of the plurality of subscribers, preventing other of
the plurality of subscribers from retrieving the message by changing the
publisher message state to a terminal state.

40. A system for message queuing among a plurality of message publishers
and a plurality of message subscribers, the system comprising:

a message queue module configured to communicate with a plurality of
message publishers and a plurality of message subscribers and to receive a data
object from a message publisher intended for at least one of a plurality of
message subscribers;

a data object processing module configured to process the data object to
create a corresponding message wherein the corresponding message includes the
data object and message information;

a publisher tracking module configured to track the status of the message
from the perspective of the message publisher;

a plurality of subscriber tracking modules configured to track the status
of the message from the perspective of the at least one of a plurality of intended
message subscribers; and

a message tracking module configured to selectively retain the message
to remain in the message queue after the message has been retrieved by the at
least one of a plurality of intended message subscribers.

41. The system of Claim 40, wherein the message queue module is further
configured to provide load balancing.

42. The system of Claim 40, wherein the message queue module is further
configured to provide subscriber pooling.

43. The system of Claim 40, wherein the message information includes a
message identifier and a message header.

44. The system of Claim 41, wherein the message header includes message
state information for the message publisher and the at least one message subscriber.

45. The system of Claim 40, wherein the message information includes message state information.

46. The system of Claim 45, wherein the message state information is stored in a log that includes time stamps for the corresponding message states.

47. A method for queuing messages received from a publisher for retrieval by a subscriber comprising:

storing messages using a queue and a queue interface module wherein the queue interface module is configurable to interface with a plurality of database protocols;

extending message storage beyond the time in which the message has been retrieved by all intended subscribers; and

tracking message states to enforce once-only delivery.

48. The method of Claim 47, wherein tracking message states includes tracking information related to message publisher states and message subscriber states.

49. The method of Claim 48, wherein the message publisher states include a "put" state, an "unavailable" state, a "revealed" state, and a "journalled" state.

50. The method of Claim 48, wherein the message publisher states include a "get" state, a "fetched" state, an "unusable" state, an "expired" state, an "inprocess" state, a "done" state, and a "journalled" state.

51. A system for queuing messages received from a publisher for retrieval by a subscriber comprising:

means for storing messages using a queue and a queue interface module wherein the queue interface module is configurable to interface with a plurality of database protocols;

means for extending message storage beyond the time in which the message has been retrieved by all intended subscribers; and

means for tracking message states to enforce once-only delivery.

52. A method for restoring message communication between a first message queue implemented using a first type of data store and a second message queue implemented using a second type of data store, the method comprising:

processing the first message queue to locate messages that have not been revealed;

for messages that have not been revealed:

5 extracting path information to find message parents of the message;

inspecting the message parents' states in the second message queue;

marking message parents as "done" and the message "revealed" if at least one of the parent messages is marked "done;"

10 marking message parents "clean" and the message "deleted" when the parent messages are not marked "done;" and

marking the message in the second message queue "revealed" if there are no parents;

recovering messages marked "clean;"

15 processing the second message queue to find messages that are marked "fetched" but not "done"; and

marking the messages that are marked "fetched" but not "done" as "clean."